**ОБЛАСТНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«ТОМСКИЙ ЭКОНОМИКО-ПРОМЫШЛЕННЫЙ КОЛЛЕДЖ»**

**ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА  
СЕРВИСА ИГРОВЫХ СОХРАНЕНИЙ «PlaySaveBack»**

***ДИПЛОМНЫЙ ПРОЕКТ***

*СПЕЦИАЛЬНОСТЬ 09.02.07 ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ПРОГРАММИРОВАНИЕ*

Выполнил:

Студент группы 1912с

\_\_\_\_\_\_\_\_\_\_\_ / Заикин В.А

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.

Проверил:

Преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_ / Уляхин В.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.

Оценка «\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

Оглавление

[Введение 3](#_Toc195676504)

[Информация о проекте 3](#_Toc195676505)

[Актуальность 3](#_Toc195676506)

[Глоссарий 4](#_Toc195676507)

[Глава 1. Техническое задание 5](#_Toc195676508)

[Общее описание 5](#_Toc195676509)

[Требования к функциональности 6](#_Toc195676510)

[Ролевая модель 7](#_Toc195676511)

[Диаграмма прецедентов 8](#_Toc195676512)

[Диаграммы последовательностей 9](#_Toc195676513)

[Технологический стек 18](#_Toc195676514)

[Хранимые данные 18](#_Toc195676515)

[Требования к REST API 19](#_Toc195676516)

[Макеты интерфейса десктопного приложения клиента 33](#_Toc195676517)

[Глава 2. Проектирование и разработка базы данных 41](#_Toc195676518)

[Концептуальная модель данных 41](#_Toc195676519)

[Логическая модель данных 42](#_Toc195676520)

[Физическая модель данных 42](#_Toc195676521)

[Словарь данных 44](#_Toc195676522)

# Введение

## Информация о проекте

|  |  |
| --- | --- |
| **Название** | Сервис игровых сохранений «PlaySaveBack» |
| **Цель** | Спроектировать и разработать сервис игровых сохранений «PlaySaveBack» |
| **Задачи** | 1. Разработать базу данных сервиса на SQLite. 2. Спроектировать структуру API сервиса. 3. Разработать сервер API сервиса на фреймворке Laravel. 4. Разработать десктопное приложение на WinUI 3 для клиента. 5. Разработать автоматизированные тесты и провести тестирование. 6. Разработать сценарии ручного тестирования десктопного приложения и провести тестирование. 7. Разработать руководство пользователя. |

## Актуальность

Современные игроки сталкиваются с проблемой хранения и управления своими игровыми сохранениями. Многие игры не предоставляют удобных средств для резервного копирования, версионности или переноса сохранений между устройствами. Это особенно актуально для пользователей, которые переустанавливают операционную систему, используют несколько игровых платформ или хотят вернуться к определённому состоянию игры. Существующие решения, такие как Steam Cloud, не всегда охватывают все игры и предоставляют ограниченные возможности управления резервными копиями.

Разрабатываемая система решает эту проблему, предоставляя удобный инструмент для хранения, управления и синхронизации игровых сохранений в облаке. Пользователи смогут загружать и восстанавливать сохранения, отслеживать изменения, а также делиться своими данными с другими игроками.

Таким образом, система повышает удобство использования сохранений, минимизирует риск потери прогресса и предоставляет новые возможности для взаимодействия пользователей с их игровыми данными.

## Глоссарий

|  |  |
| --- | --- |
| **Понятие** | **Определение понятия** |
| API | (интерфейс программирования приложений) – набор определенных правил и соглашений, который используется для взаимодействия между различными программными компонентами. API определяет, как различные части программного обеспечения могут взаимодействовать друг с другом, обычно путем обмена данными и выполнения определенных операций. |
| Postman | Инструмент для тестирования и разработки API. Он предоставляет удобный интерфейс для отправки HTTP запросов к API, а также для анализа ответов. |
| Draw.io | Бесплатный онлайн и оффлайн инструмент для создания диаграмм и схем. Он предоставляет широкий набор инструментов и элементов, которые позволяют пользователям создавать различные типы диаграмм, такие как организационные диаграммы, блок-схемы, ER-диаграммы, сетевые диаграммы, диаграммы потока процессов и многое другое. |
| CRUD | Акроним, обозначающий четыре базовые функции, используемые при работе с данными: создание, чтение, модификация, удаление. |
| Laravel | Бесплатный, открытый фреймворк для создания веб-приложений на языке программирования PHP. Он предоставляет структуру и базовые инструменты для быстрого создания веб-приложений, используя принципы модели-представления-контроллер (MVC) и другие современные подходы к разработке. |
| CRON | Планировщик задач Unix-подобных ОС, который позволяет автоматически запускать программы или скрипты в заданное время или через определённые интервалы. |
| CORS | Cross-origin resource sharing — технология современных браузеров, которая позволяет предоставить веб-страницам доступ к ресурсам другого домена. |

# Глава 1. Техническое задание

## Общее описание

Разрабатываемый сервис представляет собой систему управления сохранениями видеоигр с возможностью резервного копирования и восстановления данных. В основе проекта лежит клиентское приложение на WinUI 3, взаимодействующее с серверной частью, реализованной на Laravel. Система предназначена для геймеров, которые хотят сохранять, управлять и восстанавливать свои игровые данные без привязки к конкретному устройству.

Игры автоматически загружаются из открытого API Steam. Каждый день в полночь отправляется запрос с помощью CRON, который сверяет список игр с актуальными данными Steam, добавляя новые при их отсутствии в БД.

Система поддерживает версионность сохранений, интеграцию с облачными сервисами и возможность совместного использования сохранений с другими пользователями.

При отсутствии игры в базе данных пользователю предоставляется возможность добавить её в библиотеку в качестве сторонней. Это позволит сохранять и управлять файлами сохранений для данной игры. Однако оформление сохранений в виде публичных постов будет доступно только для игр, присутствующих в базе данных.

Функционал гостя охватывает регистрацию, авторизацию и восстановления аккаунта (см. Рисунок 1).

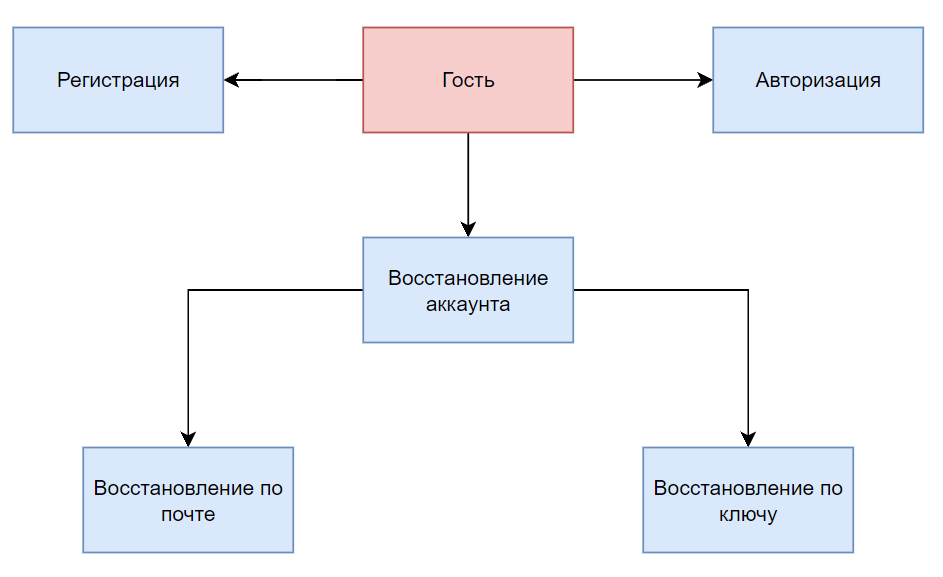


Рисунок 1 – Функционал гостя

**Функционал пользователя** охватывает управление библиотекой игр, работой с сохранениями и профилем, управление своими игровыми данными. Детали функционала пользователя представлены на Рисунок 2.

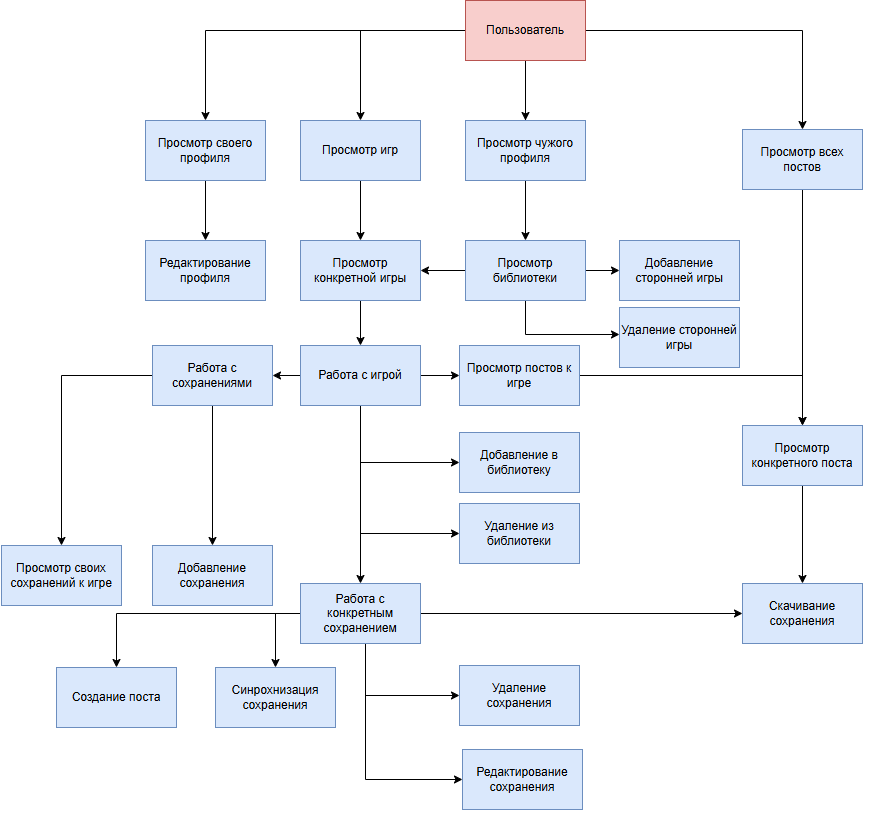


Рисунок 2 – Функционал пользователя

## Требования к функциональности

### Гости

1. Аутентификация и регистрация.
   * Аутентификация производится по токену, который генерируется при входе и отзывается при выходе.
   * Вход на другом устройстве не должен вызывать выход из аккаунта на всех устройствах одного пользователя.
2. Восстановление аккаунта
   * Возможность восстановления аккаунта по e-mail, либо по особому ключу, выданного при регистрации.

### Пользователи

1. Управление своей библиотекой игр

* Добавление игры в библиотеку из списка доступных игр.
* Добавление сторонней игры, если нужной игры нет в БД.
* Удаление игры из библиотеки.
* Добавление или удаление игры в избранное.

1. Управление сохранениями

* Просмотр списка всех сохранений.
* Просмотр сохранений для конкретной игры.
* Загрузка сохранения на облачное хранилище.
* Восстановление сохранения.
* Редактирование сохранения.
* Добавление нового сохранения для игры.
* Удаление сохранения.

1. Работа с профилем

* Просмотр своего профиля.
* Обновление данных профиля.
* Просмотр чужого профиля.

1. Работа с постами

* Создание поста
* Редактирование поста
* Просмотр публичных постов

1. Просмотр информации об играх

* Просмотр списка всех доступных игр.
* Просмотр информации о конкретной игре.

### Планировщик задач

1. Автоматическое обновление списка игр:
   * Каждый день в полночь система отправляет запрос к открытому API Steam, получает JSON и сверяет его с предыдущем.
   * Если игры отсутствуют в БД, они добавляются автоматически.

## Ролевая модель

Клиент — основной авторизированный пользователь сервиса, который загружает, организует и пользуется сохранениями.

Гость — неавторизированный пользователь, которому доступна только регистрация, вход и восстановление аккаунта.

Более структурированная и подробная информация представлена в Таблица 1.

Таблица 1 – Ролевая модель системы

|  |  |  |  |
| --- | --- | --- | --- |
| **Раздел** | **Фичи** | **Пользователь** | **Гость** |
| Аутентификация и регистрация | Вход | нет | да |
| Регистрация | нет | да |
| Восстановление аккаунта | нет | да |
| Выход | да | нет |
| Профиль | Просмотр своего профиля | да | нет |
| Просмотр чужого профиля | да | нет |
| Обновление профиля | да | нет |
| Библиотека | Получить свою библиотеку | да | нет |
| Добавить игру в библиотеку | да | нет |
| Удалить игру из библиотеки | да | нет |
| Добавить игру в избранное | да | нет |
| Убрать игру из избранного | да | нет |
| Обновление данных библиотеки | да | нет |
| Сохранения | Редактирование сохранения | да | нет |
| Просмотр своих сохранений к игре | да | нет |
| Синхронизация сохранения | да | нет |
| Создание сохранения | да | нет |
| Удаления сохранения | да(своего) | нет |
| Восстановление сохранения | да | нет |
| Скачивание сохранения | да | нет |
| Игры | Просмотр всех игр | да | нет |
| Просмотр игры | да | нет |
| Сторонние игры | Добавление сторонней игры | да | нет |
| Удаление сторонней игры | да | нет |
| Посты | Создание поста | да (к своему сохранению) | нет |
| Редактирование поста | да(своего) | нет |
| Загрузка с поста | да | нет |
| Редактирование поста | да(своего) | нет |
| Облачные сервисы | Подключение облачного сервиса Google Drive | да | нет |

## Диаграмма прецедентов

Диаграмма прецедентов используется для моделирования функциональных требований системы и взаимодействия между пользователями и системой. Она помогает определить основные сценарии использования системы и её ключевые функции. Состоит из следующих ключевых элементов:

* Акторы: роли пользователей или внешних систем, взаимодействующих с системой
* Прецеденты: функциональные возможности или сценарии, которые система предоставляет
* Связи: взаимодействия между акторами и прецедентами
* Включение и расширение: отношения между прецедентами для повторного использования или добавления дополнительного поведения

На диаграмме отражены ключевые функции, такие как управление сохранениями, библиотекой, работа со своим профилем и играми (см. Рисунок 3).

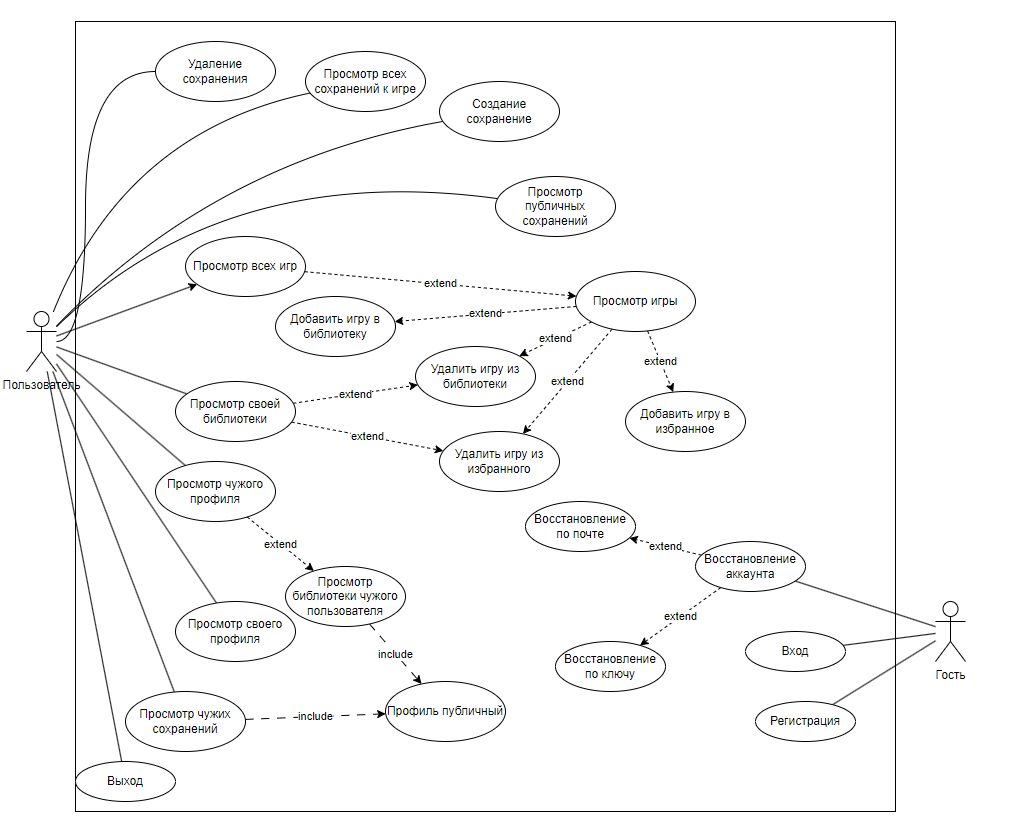


Рисунок 3 – Диаграмма прецедентов

## Диаграммы последовательностей

Диаграмма последовательностей визуализирует взаимодействие объектов в системе в рамках конкретного сценария. Она отражает последовательность сообщений, передаваемых между объектами, и их временную упорядоченность. Состоит из следующих ключевых элементов:

* Объекты и акторы: участники взаимодействия
* Сообщения: вызовы методов или передача данных между объектами
* Временная ось: вертикальная линия, показывающая порядок событий
* Активация: период активности объекта при выполнении операции
* Фреймы: обёртки над элементами для описания конкретного случая или цикла

### Общие диаграммы-дополнения

Общие диаграммы последовательности не являются самостоятельными и предназначены для дополнения к другим последовательностям для разгрузки повторяющихся элементов. Обычные диаграммы ссылаются на дополнительны посредством цветной активации по цвету выхода и припиской «(Общее: <название>)».

Валидация тела POST запроса осуществляется посредником, проверяя заданные правила на ключах (см. Рисунок 4). Имя ссылки: «валид».

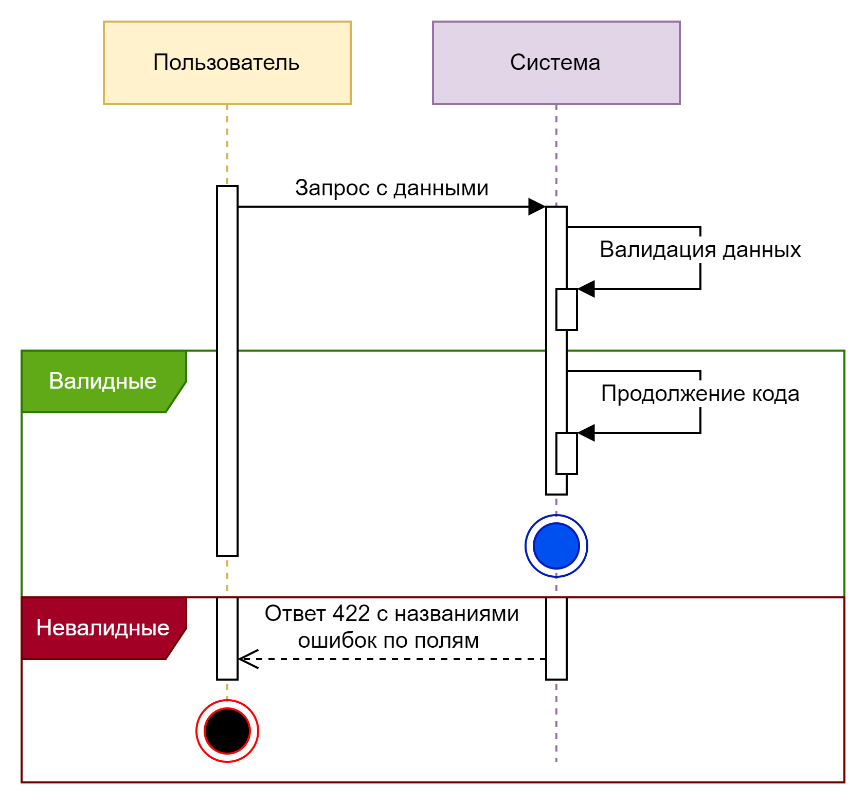


Рисунок 4 – Валидация тела запроса

Внесения в контекст кода модель пользователя и отброс неавторизированных осуществляется посредником, проверяя токен в заголовках запроса и запрашивая пользователя с этим ИД (см. Рисунок 5). Имя ссылки: «токен».

Внесения в контекст кода модель объекта по ИД и отброс если не модель не найдена осуществляется посредником, беря из маршрута ИД и запрашивая из БД её модель (см. Рисунок 6). Имя ссылки: «сущ».

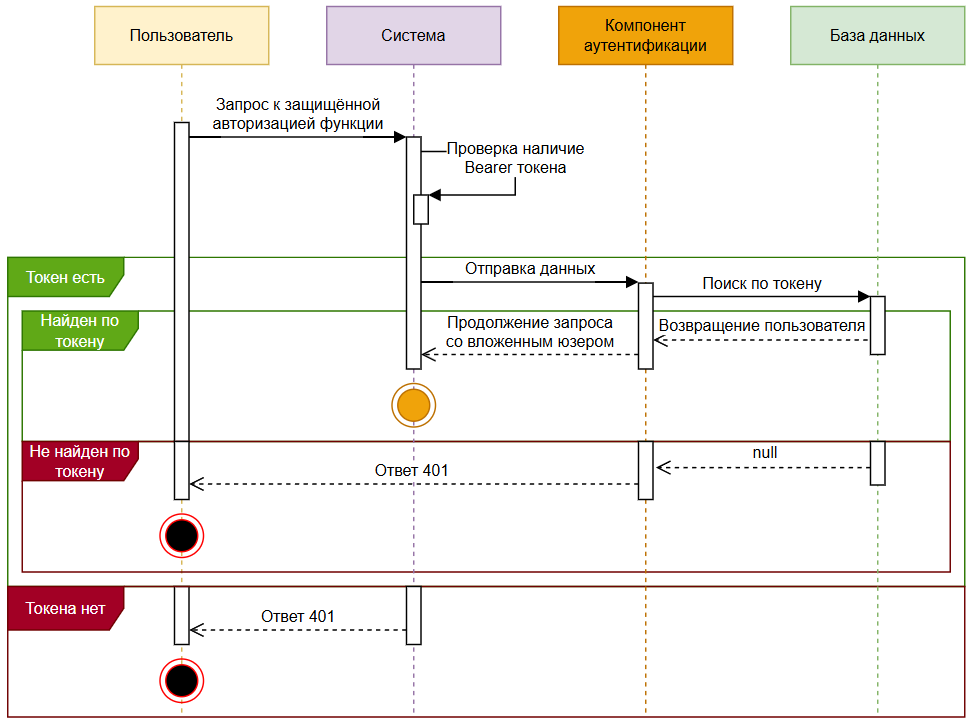


Рисунок 5 – Проверка токена и внесение в контекст пользователя

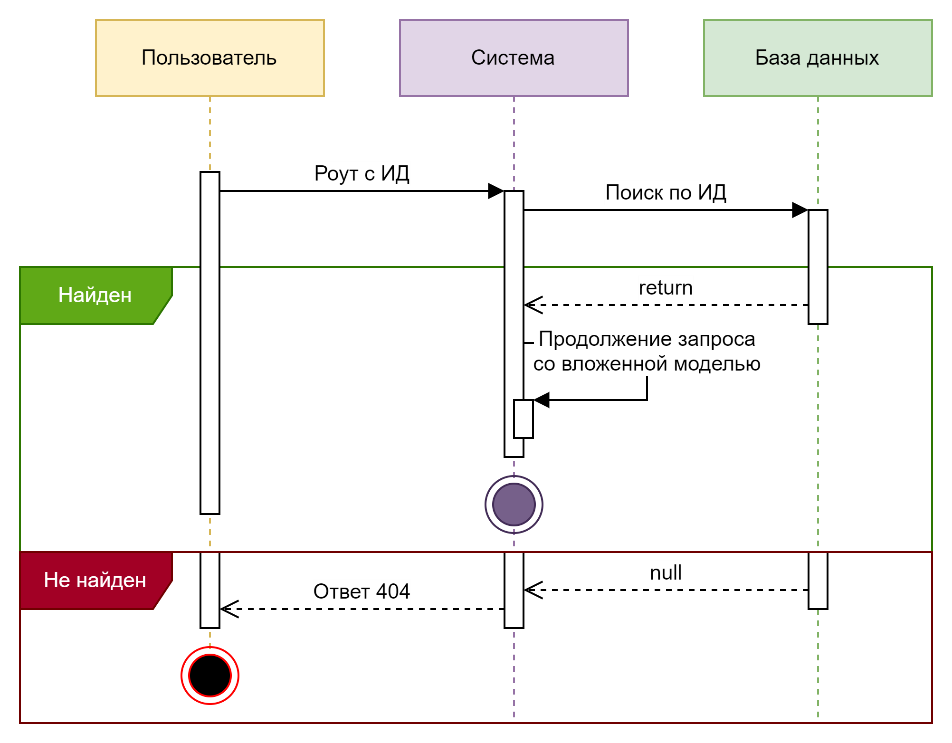


Рисунок 6 – Проверка существования объекта БД

### Функции аутентификации

Регистрация новой учётной записи представлена на Рисунок 7.

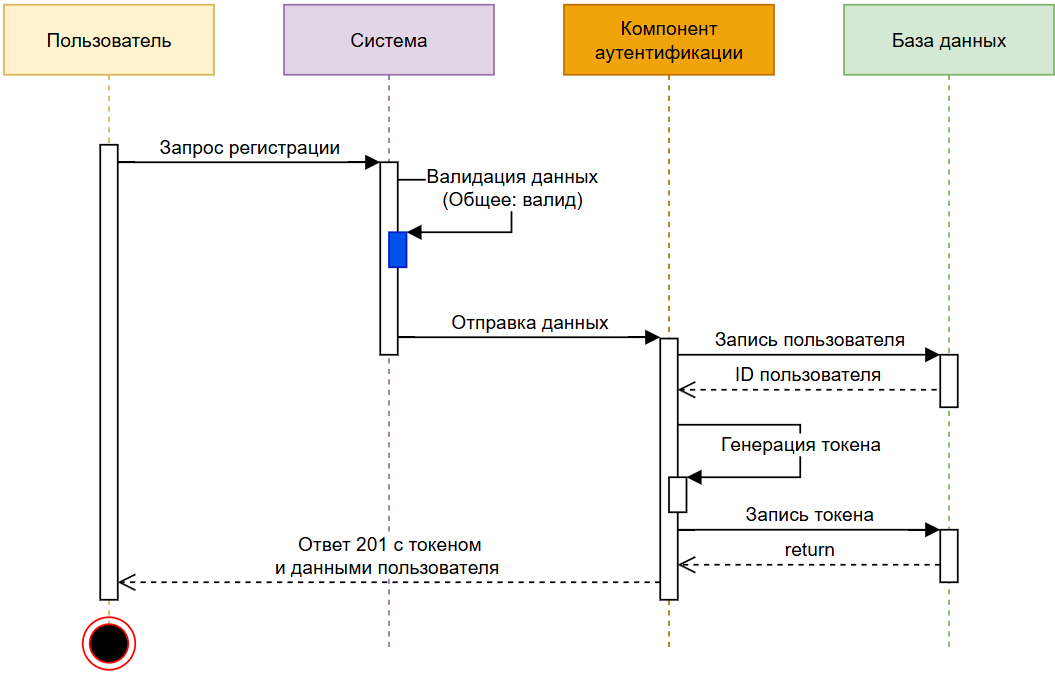


Рисунок 7 – Регистрация

Вход по реквизитам представлена на Рисунок 8.

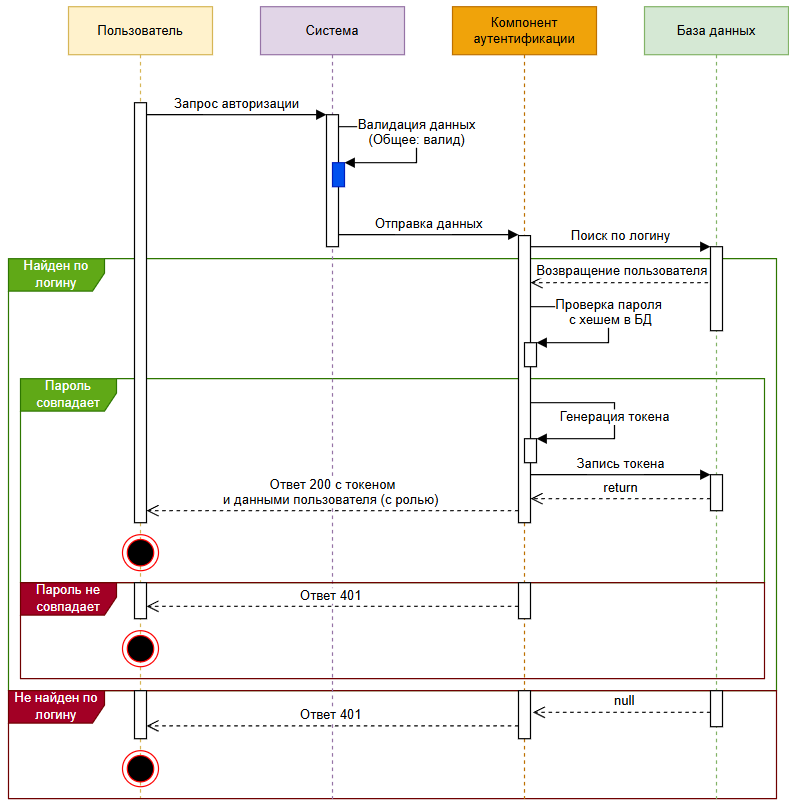


Рисунок 8 – Вход

Выход и удаление токена представлено на Рисунок 9.

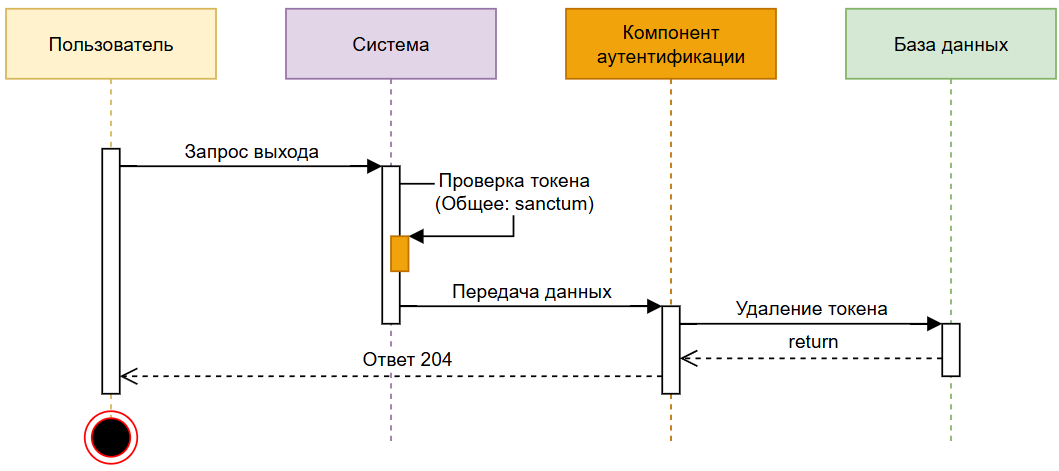


Рисунок 9 – Выход

Восстановление пароля с помощью KEY представлен на Рисунок 10.

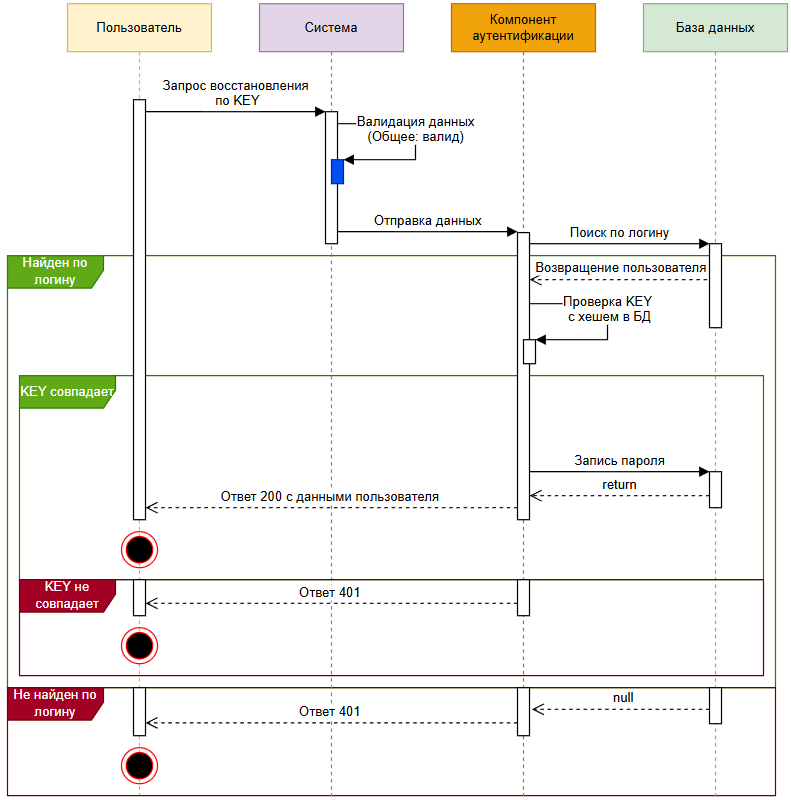


Рисунок 10 – Восстановление по KEY

### Работа с профилем

Просмотр своего профиля представлен на Рисунок 11.

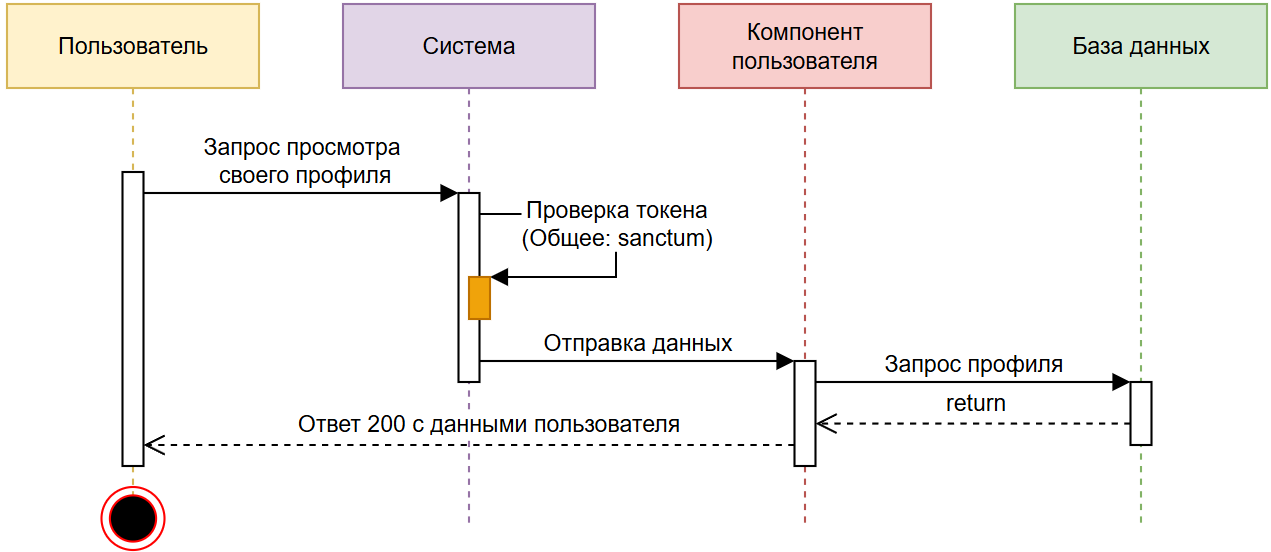


Рисунок 11 – Просмотр своего профиля

Просмотр чужого профиля представлен на Рисунок 12.

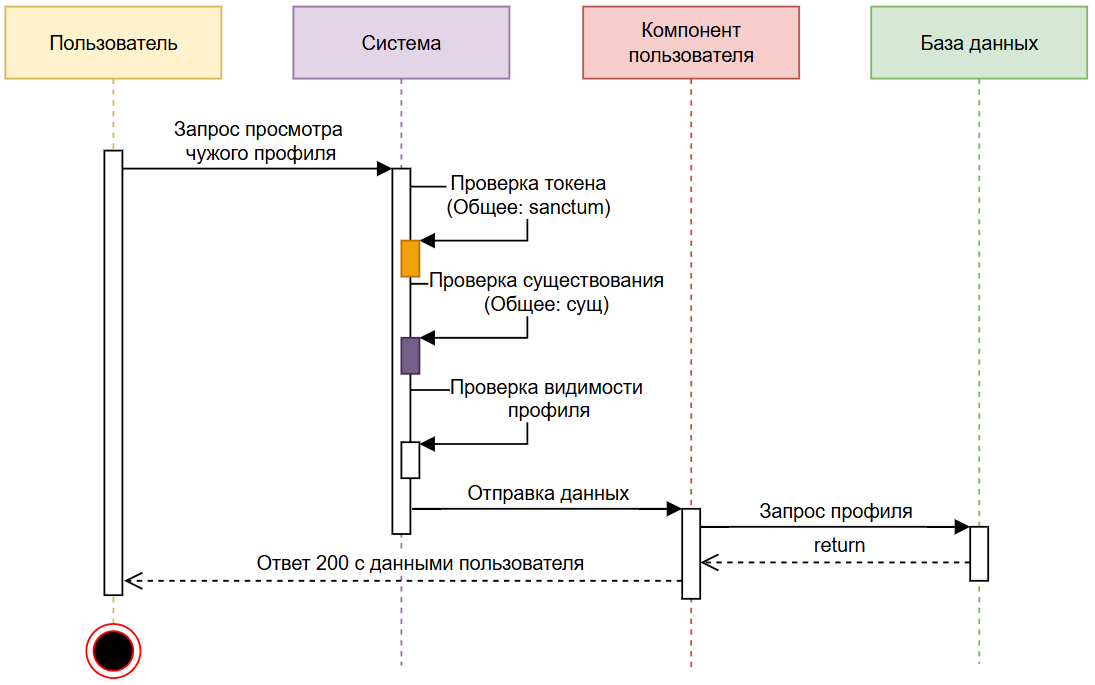


Рисунок 12 – Просмотр чужого профиля

Обновление своего профиля представлен на Рисунок 13.

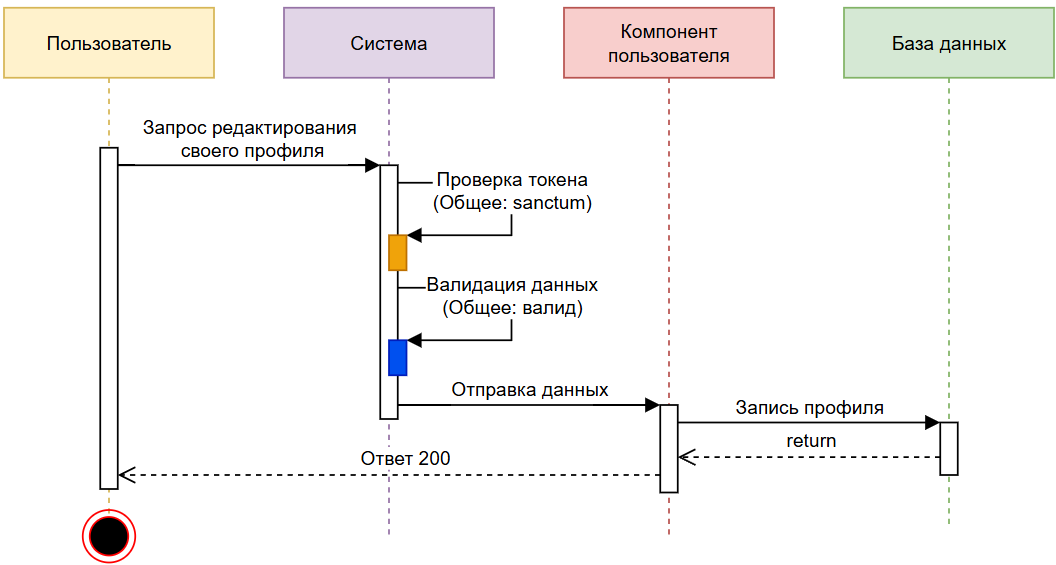


Рисунок 13 – Обновление своего профиля

### Работа с библиотекой

Получение свой библиотеки представлен на Рисунок 14.

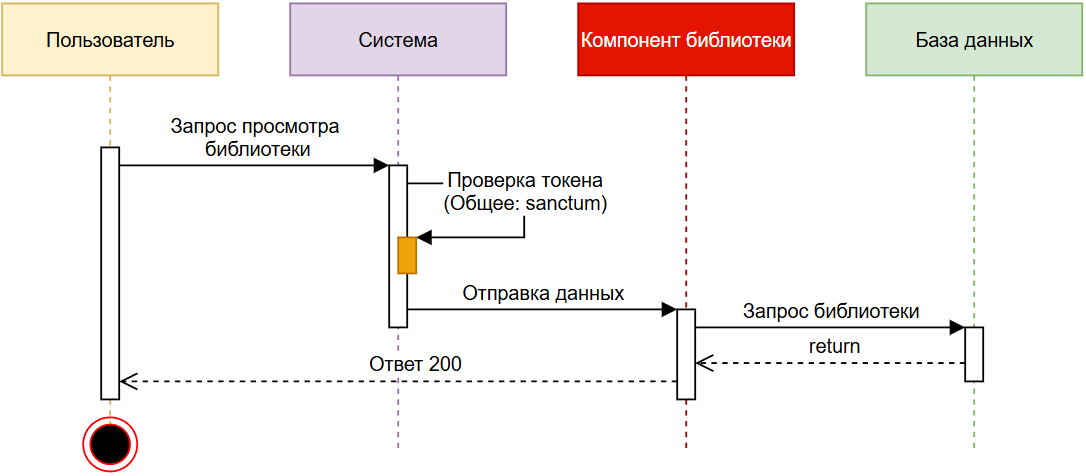


Рисунок 14 – Получение своей библиотеки

Добавление игры в библиотеку представлен на Рисунок 15.

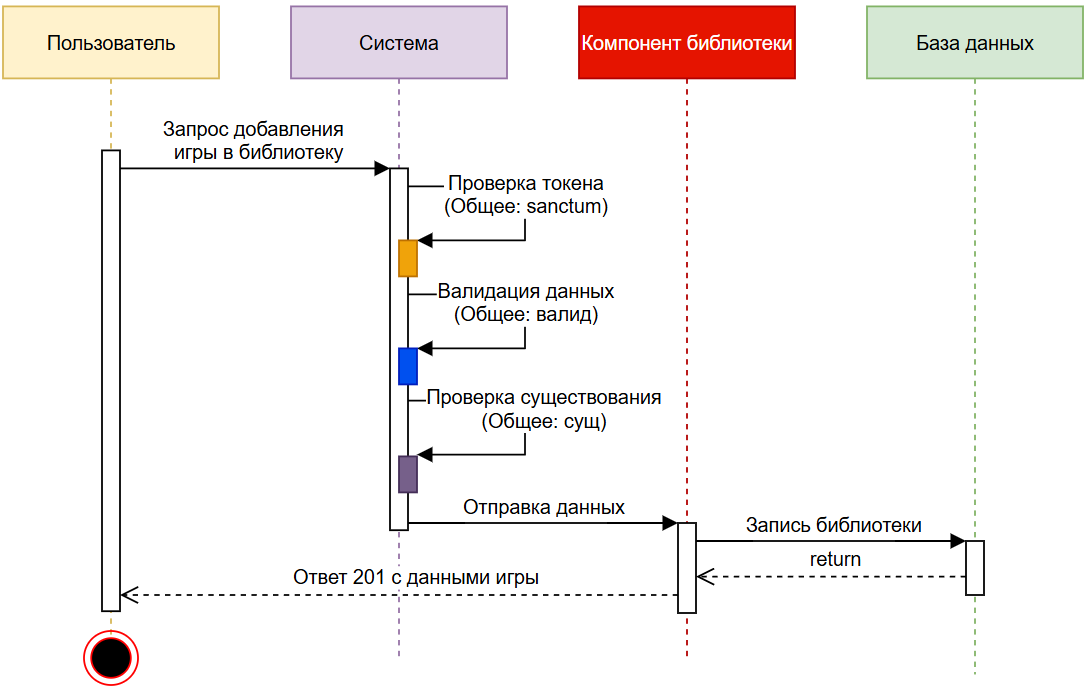


Рисунок 15 – Добавление игры в библиотеку

Удаление игры из библиотеки представлен на Рисунок 16.

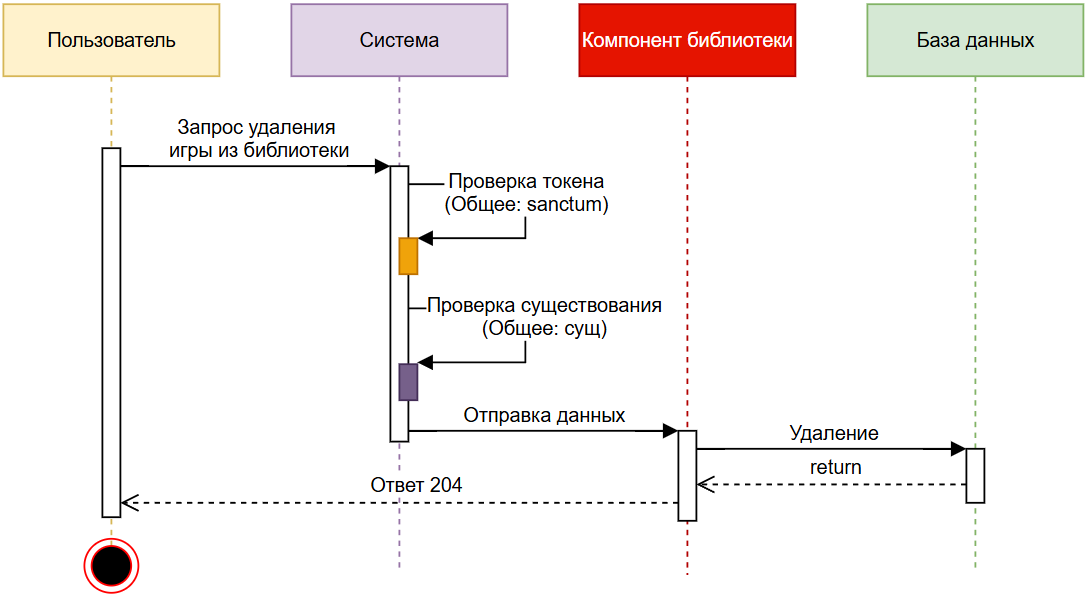


Рисунок 16 – Удаление игры из библиотеки

Обновление данных игры в библиотеки представлен на Рисунок 17.

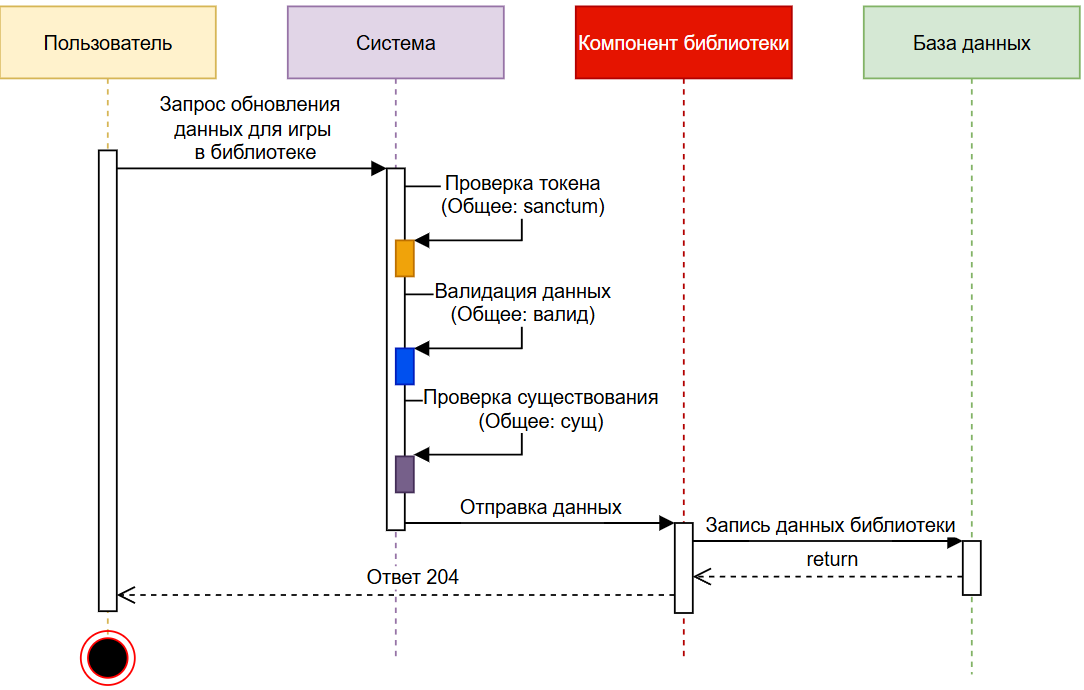


Рисунок 17 – Обновление данных игры в библиотеке

Переключение избранного статуса игры представлен на Рисунок 18.

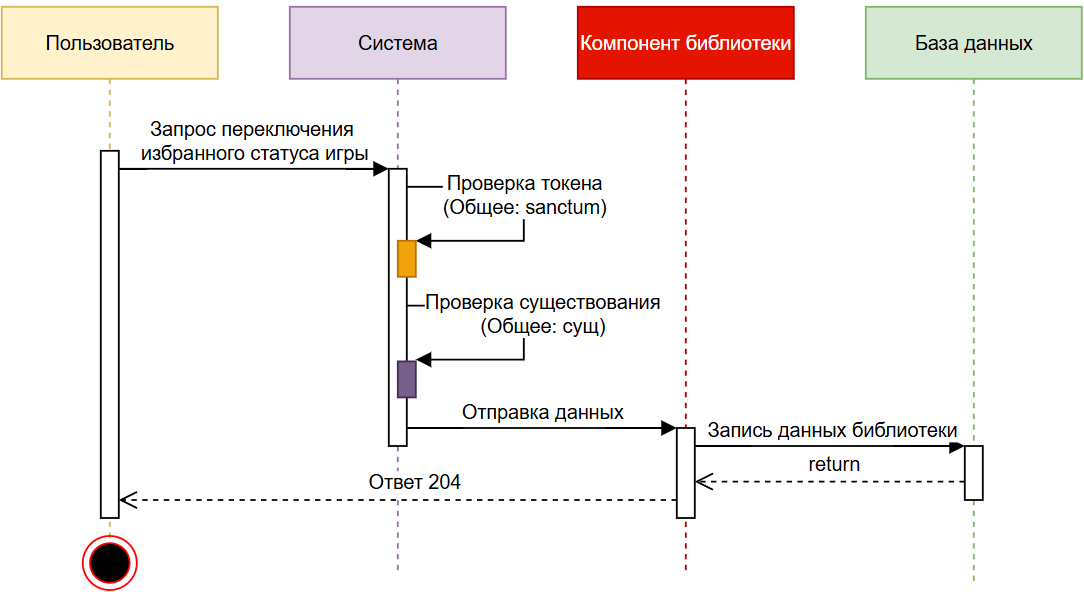


Рисунок 18 – Переключение избранного статуса игры

## Технологический стек

**Серверная часть**

Язык программирования: PHP 8.3

Фреймворк для разработки: Laravel 11

Система управление базами данными: SQLite

**Десктопное приложение**

Язык программирования и разметки: C#, XAML

Фреймворк для разработки: WinUI 3

## Хранимые данные

1. **Пользователи:** уникальный идентификатор пользователя, электронная почта, хеш пароля, никнейм, аватар, логин, заголовок профиля, ключ.
2. **Игры:** уникальный идентификатор игры, название игры, платформа, код игры.
3. **Сторонние игры:** уникальный идентификатор сторонней игры, название сторонней игры, идентификатор пользователя.
4. **Облачные сервисы:** уникальный идентификатор облачного сервиса, название облачного сервиса, описание облачного сервиса, иконка облачного сервиса.
5. **Облачные сервисы пользователей:** уникальный идентификатор связи пользователя и облачного сервиса, токен доступа, токен обновления, время истечения токена, идентификатор облачного сервиса, идентификатор пользователя.
6. **Сохранения:** уникальный идентификатор сохранения, идентификатор файла сохранения, название файла сохранения, название версии сохранения, размер файла сохранения, описание сохранения, хэш файла сохранения, дата последней синхронизации сохранения, идентификатор пользователя, идентификатор игры, идентификатор сторонней игры, идентификатор облачного сервиса пользователя.
7. **Библиотеки:** уникальный идентификатор библиотеки, идентификатор пользователя, идентификатор игры, время последнего запуска игры, время, проведённое в игре, флаг избранного, идентификатор сторонней игры.
8. **Пути игр к папке сохранений:** уникальный идентификатор пути,путь к папке сохранений, идентификатор игры.
9. **Посты:** уникальный идентификатор поста, заголовок поста, содержимое поста, идентификатор сохранения.
10. **Картинки:** уникальный идентификатор картинки, путь до картинки.
11. **Пост-Картинки:** связующий идентификатор картинки, связующий идентификатор поста.

## Требования к REST API

### Общие требования

Должно быть открытое АPI — правила CORS должны разрешать доступ с других доменов.

Идентификацию пользователя для не гостевых функции должно быть организовано посредством Bearer Token.

При попытке доступа к защищенным авторизацией функциям системы во всех запросах необходимо возвращать ответ следующего вида:

|  |
| --- |
| **Response** |
| **Status:** 401 Unauthorized  **Content-Type:** application/json  **Body:** {  “message”: “Unauthorized”  } |

При попытке доступа авторизованным пользователем к функциям недоступным для своей роли во всех запросах необходимо возвращать ответ следующего вида:

|  |
| --- |
| **Response** |
| **Status:** 403 Forbidden  **Content-Type:** application/json  **Body:** {  “message”: “Forbidden”  } |

При попытке получить несуществующий ресурс возвращать ответ следующего вида:

|  |
| --- |
| **Response** |
| **Status:** 404 Not Found  **Content-Type:** application/json  **Body:** {  “message”: “Not Found”  } |

В случае ошибок связанных с валидацией данных в теле запроса необходимо возвращать следующий ответ:

|  |
| --- |
| **Response** |
| **Status:** 422 Unprocessable Content  **Content-Type:** application/json  **Body:** {  "errors": [  <ключ>: [  <ошибка>,  <ошибка>,  ...  ],  <ключ>: [...],  ...  ]  } |

Вместо поля <ключ> должно быть название ключа запроса, в котором произошла ошибка валидации, а вместо <ошибка> — описание допущенной ошибки в этом ключе. Ошибок в одном ключе может быть множество, как и в одном запросе — множество ключей с ошибками.

### Авторизация

#### Регистрация

Запрос позволяет пользователю зарегистрироваться и вернуть его токен.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/register  **Method:** POST  **Content-Type:** application/json  **Body:** {  "name": **{nickname}**,  "login": **{login}**,  "password": **{password}**  } | **Status:** 201 Created  **Content-Type:** application/json  **Body:** {  "token": **<токен>**  "user": <**данные\_пользователя**>  }  **Возможные статус-коды:** 201, 422 |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | api | string | да |  | URL до API | https://api.pics.com |
| 2 | nickname | string | да | мин: 2, макс: 255 | Имя юзера | Testello |
| 3 | login | string | да | мин: 2, макс: 64, уник,  regex: ^[a-zA-Z0-9\_-]+$ | Логин | test |
| 4 | password | string | да | мин: 8, макс: 255 | Пароль | Test123! |

#### Авторизация

Запрос позволяет авторизировать пользователя и получить токен доступа.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/login  **Method:** POST  **Content-Type:** application/json  **Body:** {  "login": **{login}**  "password": **{password}**  } | **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "token":<**токен**>  "user": {  <**"role":** **"admin"**>  "name": "Testello",  "login": "test"  }  }  **Возможные статус-коды:** 200, 401, 422 |

Входные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Описание** | **Пример** |
| 1 | login | string | да | Логин | mylogin |
| 2 | password | string | да | Пароль | Pass123! |

#### Выход

Запрос предназначен для удаления текущего токена пользователя.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/logout  **Method:** POST  **Authorization:** Bearer **{token}** | **Status:** 204 No Content  **Возможные статус-коды:** 200, 401 |

Входные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Описание** | **Пример** |
| 1 | token | string | да | API токен | 27|fxR7b3H3o5nPYAdtWhbk61do |

### Работа с сохранениями

#### Создание сохранения

Запрос для создания нового личного альбома пользователя. Возвращает объект альбома с его ID в БД и создаёт папку с выведенным названием в ФС.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums  **Method:** POST  **Authorization:** Bearer **{token}**  **Content-Type:** application/json  **Body:** {  "**name**": **{name}**,  "path": **{path}**  } | **Status:** 201 Created  **Content-Type:** application/json  **Body:** {  "id": <**id\_альбома**>,  "name": **{name}**,  "path": **{path}**  }  **Возможные статус-коды:** 201, 401, 409, 422 |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | name | string | да | макс: 255, уник(для юзера) | Название | Камера |
| 2 | path | string | нет | макс: 255, уник(для юзера) | Путь на клиенте для синхронизации | DCIM/Camera |

#### Получение альбома

Запрос для удаления альбома. Удаляет все связанные данные в БД и все картинки в ФС.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**  **Method:** GET  **Authorization:** Bearer **{token}** | **Status:** 200 OK  **Content-Type:** application/json  **Body:**  Свой  {  "album": {  "id": 1,  "name": "Камера",  "createdAt": "2024-12-16T10:13:23",  "grantAccesses": [  {  "id": 215,  "name": "Чел"  }, { ... }, { ... }  ],  "picturesCount": 31,  "picturesInfo": {  "sign": "2\_JDJ5JDEyJG11RGc24aWRX",  "ids": [84, 30, 69, 59]  }  }  }  Чужой доступный  {  "album": {  "id": 1,  "name": "Скаченное",  "owner": {  "id": 215,  "name": "Чел"  },  "picturesCount": 29,  "picturesInfo": {  "sign": "2\_JDJ5JDEyJHoMVpWaEJjVzNSV2",  "ids": [49, 41, 30, 87]  }  }  }  **Возможные статус-коды:** 200, 401, 403, 404 |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | albumId | integer | да | существовать | Код альбома | 87 |

#### Удаление сохранения

Запрос для удаления альбома. Удаляет все связанные данные в БД и все картинки в ФС.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**  **Method:** DELETE  **Authorization:** Bearer **{token}** | **Status:** 204 Deleted  **Возможные статус-коды:** 204, 401, 403, 404 |

#### Получение списка сохранений

Запрос для получения списка альбомов. Возвращает список всех альбомов пользователя и доступных из БД.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums  **Method:** GET  **Authorization:** Bearer **{token}** | **Status:** 200 OK  **Content-Type:** application/json  **Body:**  {  "own": [  {  "id": 354,  "name": "Камера",  "path": "DCIM/Camera",  "createdAt": "2024-12-16T10:13:23",  "grantAccessesCount": 3,  "picturesCount": 31,  "picturesInfo": { ... }  }, { ... }, { ... }  ],  "accessible": [  {  "id": 5372,  "name": "Скаченное",  "path": "Pictures/Downloads",  "createdAt": "2024-12-16T10:13:23",  "owner": {  "id": 215,  "name": "Чел"  },  "picturesCount": 31,  "picturesInfo": { ... }  }, { ... }, { ... }  ]  }  **Возможные статус-коды:** 200, 401 |

### Работа с играми

#### Добавление игры

Запрос для загрузки на ФС сервера картинок и информации об каждой в БД

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**/pictures  **Method:** POST  **Authorization:** Bearer **{token}**  **Content-Type:** multipart/form-data  **Body:** {  "pictures[]": **{pictures}**,  } | **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "successful": [  {  "id": 1,  "name": "test.jpg",  "date": "2024-10-29 19:56:10",  "size": 1696738,  "width": 1360,  "height": 2128,  “hash”: "df59b7d94c03b59a”  }, { ... }  ],  "errored": [  {  "name": "image.png",  "message": “already exist"  }, { ... }  ]  }  **Возможные статус-коды:** 200, 401, 403, 404 |

Входные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Условия** | **Описание** |
| 1 | pictures | array | да | мин: 1 | Массив картинок |
| 1 | pictures.\* | file | нет | mimes: jpeg,png,gif | Картинка |

#### Получение списка игр

Запрос для получения списка всех картинок в указанном альбоме. Так же выводится временная (на сутки) сигнатура доступа для получения превью/оригинала/загрузки.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**/pictures  **Method:** GET  **Authorization:** Bearer **{token}** | **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "sign": <**сигнатура**>  "pictures": [  {  "id": 1,  "name": "test.jpg",  "date": "2024-10-29 19:56:10",  "size": 1696738,  "width": 1360,  "height": 2128,  “hash”: "df59b7d94c03b59a”  },  { ... }  ],  }  **Возможные статус-коды:** 200, 401, 403, 404 |

#### Отображение конкретной игры

Запрос для получения файла-превью указанного размера (разрешённый размер из публичной информации о сервере). Для аутентификация используется сигнатура доступа, которую можно получить из запроса списка картинок. Если указать запрещённый размер, то будет перенаправление на приближённый размер.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**  /pictures/**{pictureId}**  /thumb/**{orient}{size}**  ?sign=**{sign}**  **Method:** GET | **Status:** 200 OK  **Content-Type:** image/jpeg  **Возможные статус-коды:** 200, 301, 403, 404 |

Входные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Описание** | **Пример** |
| 1 | pictureId | integer | да | Код картинки | 529 |
|  | orient | "w"|"h"|"q" |  | Ось указания размера (по ширине, высоте или обрезка в квадрат) |  |
| 2 | size | integer | да | Размер по оси (пиксели) | 720 |
| 3 | sign | string | да | Сигнатура доступа | 2\_5EyJHc3aBoY1Ju |

#### Отображение данных игры

Запрос для отображения файла. Для аутентификация используется сигнатура доступа, которую можно получить из запроса списка картинок.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**  /pictures/**{pictureId}**  /original  ?sign=**{sign}**  **Method:** GET | **Status:** 200 OK  **Content-Type:** image/<"jpeg"|"png"|...>  **Возможные статус-коды:** 200, 403, 404 |

#### Скачивание игры

Запрос для скачивания файла. Для аутентификация используется сигнатура доступа, которую можно получить из запроса списка картинок.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**  /pictures/**{pictureId}**  /download  ?sign=**{sign}**  **Method:** GET | **Status:** 200 OK  **Content-Type:** image/<"jpeg"|"png"|...>  **Content-Disposition:**  attachment; filename=<имя\_файла>  **Возможные статус-коды:** 200, 403, 404 |

#### Удаление игры

Запрос для удаления картинки. Удаляет связанные данные в БД и картинку в ФС.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**  /pictures/**{pictureId}**  **Method:** DELETE  **Authorization:** Bearer **{token}** | Успех  **Status:** 204 Deleted  **Возможные статус-коды:** 204, 401, 403, 404 |

### Работа с библиотекой

#### Добавление игры в библиотеку

Запрос для создания личного тега пользователя.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums  **Method:** POST  **Authorization:** Bearer **{token}**  **Content-Type:** application/json  **Body:** {  "value": **{value}**  } | **Status:** 201 Created  **Content-Type:** application/json  **Body:** {  "id": <**id\_тега**>,  "name": <**название**>  }  **Возможные статус-коды:** 201, 401, 409 |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | value | string | да | макс: 255, уник(для юзера) | Название | Семья |

#### Удаление игры из библиотеки

Запрос на получение списка личных тегов.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums  **Method:** GET  **Authorization:** Bearer **{token}** | **Status:** 200 OK  **Content-Type:** application/json  **Body:** [  {  "id": <**id\_тега**>,  "name": <**название**>  },  { ... },  ]  **Возможные статус-коды:** 204, 401 |

#### Добавление игры в избранное

Запрос для добавления связи тега с картинкой.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/{**albumId**}  /pictures/**{pictureId}**  /tags/**{tagId}**  **Method:** POST  **Authorization:** Bearer **{token}** | **Status:** 204 No Content  **Возможные статус-коды:** 204, 401, 403, 404, 409 |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | tagId | string | да | существовать | Код тега | 5 |

#### Удаление игры из избранного

Запрос для удаления связи тега с картинкой.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/{**albumId**}  /pictures/**{pictureId}**  /tags/**{tagId}**  **Method:** DELETE  **Authorization:** Bearer **{token}** | **Status:** 204 No Content  **Возможные статус-коды:** 204, 401, 403, 404, 409 |

### Работа с облачными сервисами

#### Просмотр списка облаычных сервисов

Запрос для добавления предупреждения на пользователя.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/users/**{user}**/warnings  **Method:** POST  **Authorization:** Bearer **{token}**  **Content-Type:** application/json  **Body:** {  "comment": **{description}**  } | **Status:** 201 Created  **Возможные статус-коды:** 201, 401, 403, 404 |

Входные параметры:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Описание** |
| 1 | comment | string | нет | Пояснение |

#### Подключение облачного сервиса

Запрос для удаления предупреждения с пользователя.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/users/**{user}**/warnings/**{warnings}**  **Method:** DELETE  **Authorization:** Bearer **{token}** | **Status:** 204 No Content  **Возможные статус-коды:** 204, 401, 403, 404 |

### Работа с профилем

#### Получение себя

Запрос на данных о себе.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/users/me  **Method:** GET  **Authorization:** Bearer **{token}** | **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "user": {  "id": 1,  "name": "Testello",  "login": "test",  <"role": "admin">  }  }  **Возможные статус-коды:** 200, 401 |

#### Редактирование себя

Запрос на изменение данных о себе.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/users/me  **Method:** POST  **Authorization:** Bearer **{token}**  **Content-Type:** application/json  **Body:** {  "name": **{nickname}**,  "login": **{login}**,  "password": **{password}**  } | **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "user": {  "id": 1,  "name": "Testello",  "login": "test",  <"role": "admin">  }  }  **Возможные статус-коды:** 200, 401, 422 |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | nickname | string | нет | мин: 2, макс: 255 | Имя юзера | Testello2 |
| 2 | login | string | нет | мин: 2, макс: 64, уник,  regex: ^[a-zA-Z0-9\_-]+$ | Логин | test2 |
| 3 | password | string | нет | мин: 8, макс: 255 | Пароль | NewPass! |

#### Просмотр списка пользователей

Запрос на просмотр списка пользователей.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/users  **Method:** GET  **Authorization:** Bearer **{token}** | **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "users": [  {  "id": 2,  "name": "Чел",  "isBanned": false,  "login": "test1",  <"role": "admin">  "complaintsAboutCount": 21,  "complaintsAboutAcceptedCount": 0,  "complaintsFromCount": 18,  "complaintsFromAcceptedCount": 9,  "albumsCount": 1,  "albumsViaAccessCount": 2,  "picturesCount": 31  }, { ... }, { ... }  ]  }  **Возможные статус-коды:** 200, 401, 403 |

#### Просмотр пользователя

Запрос просмотра пользователя.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/users/**{userId}**  **Method:** GET  **Authorization:** Bearer **{token}** | **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "user": {  "id": 2,  "name": "Чел",  "isBanned": false,  "login": "test1",  <"role": "admin">  "complaintsAbout": [  {  "id": 1,  "status": null,  "description": "Плохие картинки"  }, { ... }, { ... }  ],  "complaintsFrom": [  {  "id": 1,  "status": null,  "description": "Мне не нравится"  }, { ... }, { ... }  ],  "albumsCount": 1,  "albumsViaAccessCount": 2,  "picturesCount": 31,  "warnings": [  {  "id": 1,  "comment": "Больше так не делай"  }, { ... }, { ... }  ],  }  }  **Возможные статус-коды:** 200, 401, 403, 404 |

#### Редактирование пользователя

Запрос редактирование пользователя.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/users/**{userId}**  **Method:** POST  **Authorization:** Bearer **{token}**  **Content-Type:** application/json  **Body:** {  "name": **{nickname}**,  "login": **{login}**,  "password": **{password},**  "role\_id": **{roleId}**,  "is\_banned": **{**isBanned**},**  } | **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "user": {  "id": 2,  "name": "Чел",  "isBanned": false,  "login": "test1",  <"role": "admin">  }  }  **Возможные статус-коды:** 200, 401, 403, 404 |

Входные параметры:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Условия** | **Описание** | **Пример** |
| 1 | roleId | integer | нет | существовать | ИД роль | 2 |
| 2 | isBanned | boolean | нет |  | Статус блокировки | true |

### Работа с постами

#### Просмотр постов

Запрос публичных настроек.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/  **Method:** GET | **Status:** 200 OK  **Content-Type:** application/json  **Body:** {  "settings": {  "allowed\_upload\_mimes": [  "jpeg", "jpg", "png", "gif"  ],  "allowed\_preview\_sizes": [  144, 240, 360, 480, 720, 1080  ],  "warning\_limit\_for\_ban": 3,  "free\_storage\_limit": 5368709120,  "complaint\_types": [  {  "id": 5,  "name": "Нарушение авторских прав"  },  {  "id": 6,  "name": "Шок-контент"  }  ],  "is\_upload\_disabled": true  }  }  **Возможные статус-коды:** 200 |

#### Просмотр конкретного поста

Запрос для получения файла-превью указанного размера (разрешённый размер из публичной информации о сервере). Для аутентификация используется сигнатура доступа, которую можно получить из запроса списка картинок. Если указать запрещённый размер, то будет перенаправление на приближённый размер.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**  /pictures/**{pictureId}**  /thumb/**{orient}{size}**  ?sign=**{sign}**  **Method:** GET | **Status:** 200 OK  **Content-Type:** image/jpeg  **Возможные статус-коды:** 200, 301, 403, 404 |

Входные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Описание** | **Пример** |
| 1 | pictureId | integer | да | Код картинки | 529 |
|  | orient | "w"|"h"|"q" |  | Ось указания размера (по ширине, высоте или обрезка в квадрат) |  |
| 2 | size | integer | да | Размер по оси (пиксели) | 720 |
| 3 | sign | string | да | Сигнатура доступа | 2\_5EyJHc3aBoY1Ju |

#### Создание поста

Запрос для получения файла-превью указанного размера (разрешённый размер из публичной информации о сервере). Для аутентификация используется сигнатура доступа, которую можно получить из запроса списка картинок. Если указать запрещённый размер, то будет перенаправление на приближённый размер.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**  /pictures/**{pictureId}**  /thumb/**{orient}{size}**  ?sign=**{sign}**  **Method:** GET | **Status:** 200 OK  **Content-Type:** image/jpeg  **Возможные статус-коды:** 200, 301, 403, 404 |

Входные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Описание** | **Пример** |
| 1 | pictureId | integer | да | Код картинки | 529 |
|  | orient | "w"|"h"|"q" |  | Ось указания размера (по ширине, высоте или обрезка в квадрат) |  |
| 2 | size | integer | да | Размер по оси (пиксели) | 720 |
| 3 | sign | string | да | Сигнатура доступа | 2\_5EyJHc3aBoY1Ju |

#### Редактирование поста

Запрос для получения файла-превью указанного размера (разрешённый размер из публичной информации о сервере). Для аутентификация используется сигнатура доступа, которую можно получить из запроса списка картинок. Если указать запрещённый размер, то будет перенаправление на приближённый размер.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**  /pictures/**{pictureId}**  /thumb/**{orient}{size}**  ?sign=**{sign}**  **Method:** GET | **Status:** 200 OK  **Content-Type:** image/jpeg  **Возможные статус-коды:** 200, 301, 403, 404 |

Входные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Описание** | **Пример** |
| 1 | pictureId | integer | да | Код картинки | 529 |
|  | orient | "w"|"h"|"q" |  | Ось указания размера (по ширине, высоте или обрезка в квадрат) |  |
| 2 | size | integer | да | Размер по оси (пиксели) | 720 |
| 3 | sign | string | да | Сигнатура доступа | 2\_5EyJHc3aBoY1Ju |

#### Удаление поста

Запрос для получения файла-превью указанного размера (разрешённый размер из публичной информации о сервере). Для аутентификация используется сигнатура доступа, которую можно получить из запроса списка картинок. Если указать запрещённый размер, то будет перенаправление на приближённый размер.

|  |  |
| --- | --- |
| **Request** | **Response** |
| **URL: {api}**/albums/**{albumId}**  /pictures/**{pictureId}**  /thumb/**{orient}{size}**  ?sign=**{sign}**  **Method:** GET | **Status:** 200 OK  **Content-Type:** image/jpeg  **Возможные статус-коды:** 200, 301, 403, 404 |

Входные параметры:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Тип** | **Обяз** | **Описание** | **Пример** |
| 1 | pictureId | integer | да | Код картинки | 529 |
|  | orient | "w"|"h"|"q" |  | Ось указания размера (по ширине, высоте или обрезка в квадрат) |  |
| 2 | size | integer | да | Размер по оси (пиксели) | 720 |
| 3 | sign | string | да | Сигнатура доступа | 2\_5EyJHc3aBoY1Ju |

## Макеты интерфейса десктопного приложения клиента

В рамках данного проекта используются схематические макеты (wireframe), определяющие расположение элементов интерфейса и логику пользовательского взаимодействия. Дизайн и стилизация не проработаны на данном этапе.

### Окно входа

Окно входа представляет собой стартовый интерфейс системы, который является первым и основным пунктом взаимодействия пользователя с приложением — оно автоматически открывается при первоначальном запуске программы или в случае, когда пользователь явно завершил предыдущую сессию, нажав кнопку "Выйти". Главная функциональная нагрузка данного окна заключается в обеспечении безопасной и удобной процедуры аутентификации, когда пользователь подтверждает свою личность путём ввода заранее созданных учётных данных, а именно — уникального логина или привязанного email-адреса в сочетании с секретным паролем (см. Рисунок 19). Помимо базового сценария входа, интерфейс предусматривает вспомогательные опции, призванные охватить все возможные пользовательские ситуации: для тех, кто утратил доступ к своему аккаунту, предусмотрен механизм восстановления пароля на выбор пользователя (см. Рисунок 20), а для новых пользователей, только знакомящихся с приложением, доступна интуитивно понятная возможность мгновенного перехода к процессу регистрации, где можно создать совершенно новый аккаунт и получить полный доступ ко всем функциям системы.

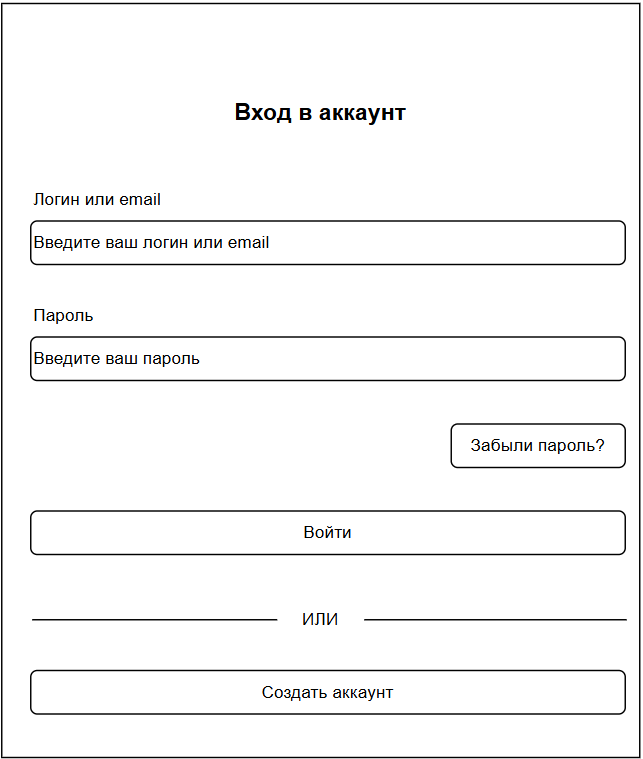


Рисунок 19 – Окно входа

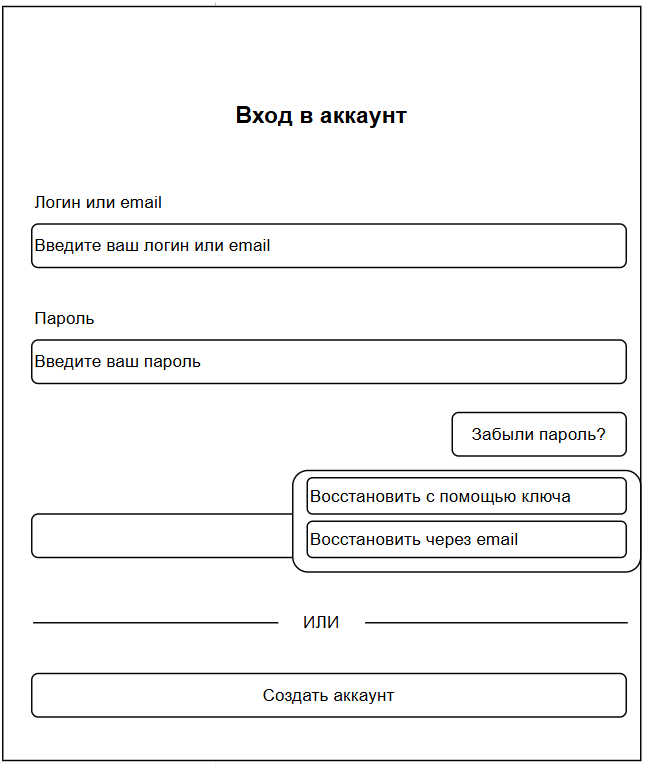


Рисунок 20 – Выбор восстановления пароля

### Страница восстановления пароля по ключу

Окно восстановления пароля по ключу является частью системы безопасности приложения и позволяет пользователю сбросить утраченный или забытый пароль (см. Рисунок 21). На этот экран можно попасть из окна входа, кликнув на ссылку *«Забыли пароль?»* под полем ввода пароля. После успешного восстановления пользователь перенаправляется обратно на экран входа, где может авторизоваться с новыми данными.

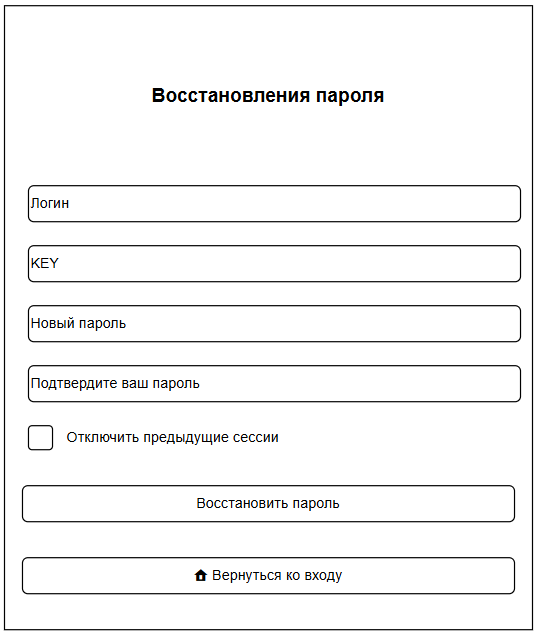


Рисунок 21 – Восстановление пароля по ключу

### Окно регистрации

Окно регистрации нового аккаунта выполняет две основные функции: сбор необходимых данных для идентификации пользователя (логин, email и пароль) и обеспечение безопасности будущего аккаунта через многоуровневую проверку вводимой информации (см. Рисунок 22).

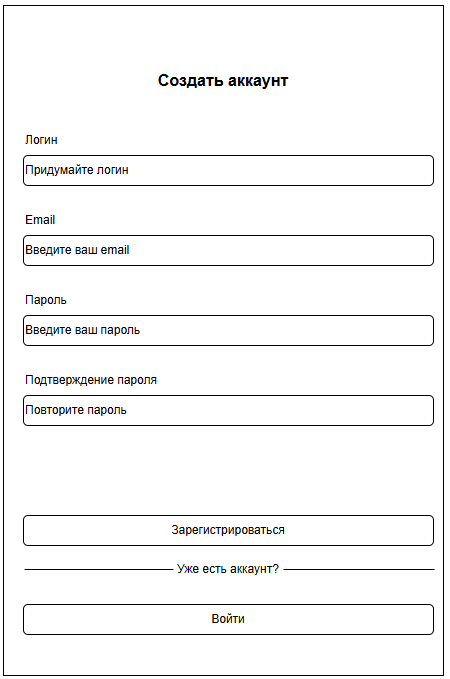


Рисунок 22 – Окно регистрации

После успешного заполнения формы система генерирует уникальный KEY для восстановления доступа - критически важный элемент защиты, который пользователь должен сохранить в надёжном месте. Особенностью процесса является двухэтапный подход: сначала стандартная регистрация с базовыми данными, а затем - обязательное подтверждение сохранения резервного KEY (см. Рисунок 23), что значительно снижает риски безвозвратной потери доступа к аккаунту.

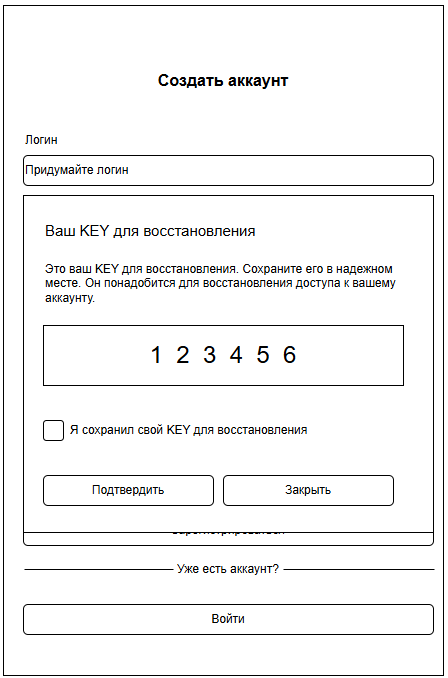


Рисунок 23 – Подтверждение KEY

### Главное окно приложения

Главное окно является центральным узлом взаимодействия пользователя с приложением после успешной авторизации. Оно предоставляет доступ ко всем ключевым функциям через интуитивную навигационную панель и рабочую область для отображения контента (см. Рисунок 24).

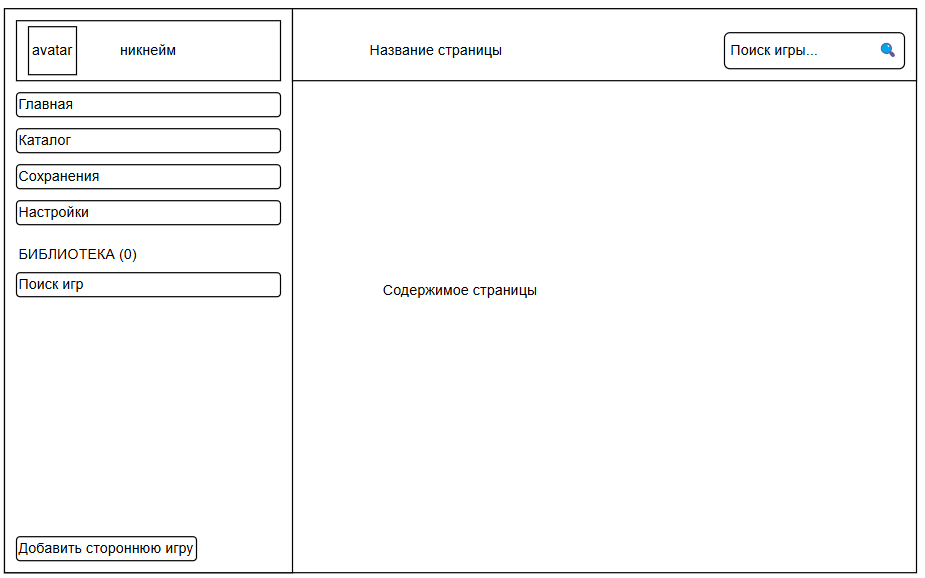


Рисунок 24 – Главное окно приложения

### Страница каталога

Страница каталога игр является центральным разделом приложения, предназначенным для просмотра, поиска и выбора игр (см. Рисунок 25). Оно открывается при переходе из главного меню по соответствующей вкладке и предоставляет пользователю удобный инструментарий для навигации по обширному ассортименту контента.

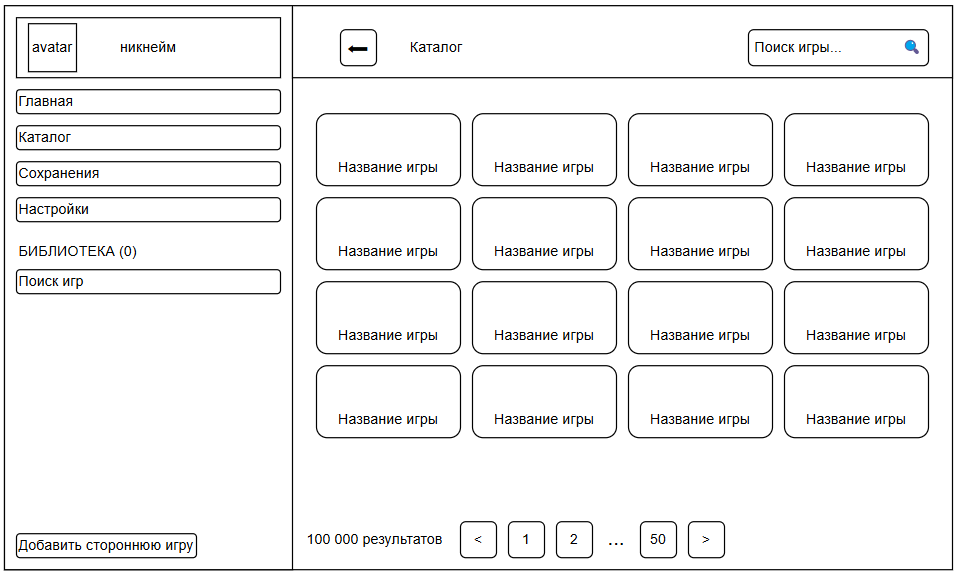


Рисунок 25 – Страница каталога

### Страница игры

Страница игры открывается при переходе из раздела *Каталог* по нажатию карточки с игрой (см. Рисунок 25) и служит для отображения полной информации о конкретном продукте. Это ключевая точка взаимодействия пользователя с контентом перед запуском или добавлением в библиотеку (см. ).

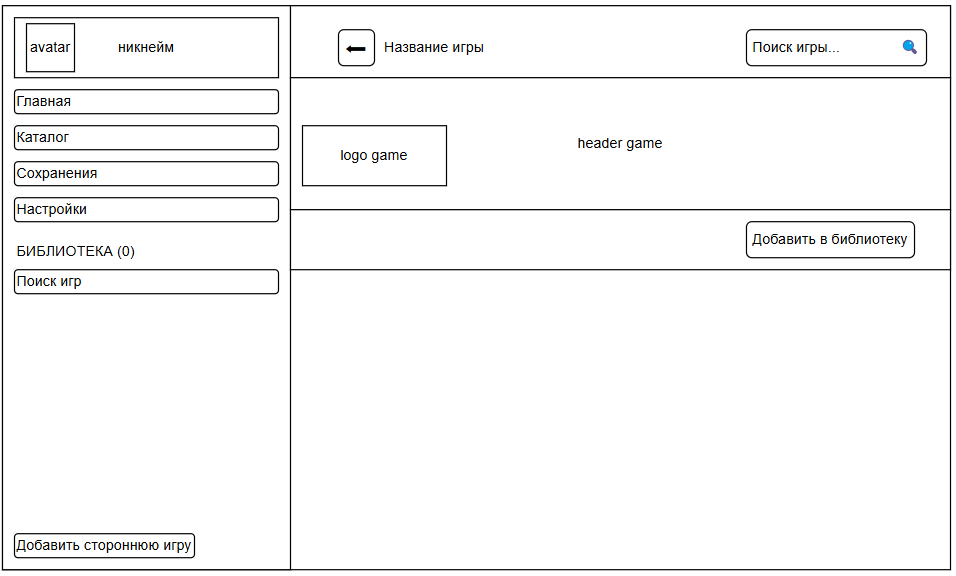


Рисунок 26 – Страница игры

Страница игры после добавления в библиотеку становится личной игровой панелью пользователя, предоставляя доступ к управлению игрой, отслеживанию статистики и работе с сохранениями, последующее отображение страницы игры при переходе из библиотеки или каталога изображено на Рисунок 27.



Рисунок 27 – Страница игры после добавления в библиотеку

# Глава 2. Проектирование и разработка базы данных

## Концептуальная модель данных

Концептуальная модель — это формализованное описание предметной области, не связанное с какими-либо компьютерными средствами. Сущности (таблицы) обозначаются прямоугольниками; Атрибуты (поля таблиц) — овалы, которые могут быть объедены в скруглённые прямоугольники; Связи с обозначением мощности — стрелками (в направлении, где мощность равна многим — двойная стрелка, а со стороны, где она равна единице — одинарная).

Анализируя предметную область, можно выделить следующие сущности с их атрибутами (см. Рисунок 34):

1. Сущность «**Картинка**» с атрибутом «Путь».
2. Сущность «**Пост**» с атрибутами «Заголовок», «Контент».
3. Сущность «**Пользователь**» с атрибутами «Логин», «Никнейм», «Хеш пароля», «Аватар», «Шапка», «Хеш ключа», «Почта».
4. Сущность «**Путь**» с атрибутом «Путь».
5. Сущность «**Игра**», c атрибутами «Платформа», «Название», «Код».
6. Сущность «**Сторонняя игра**» с атрибутом «Название».
7. Сущность «**Облачный сервис**» с атрибутами «Название», «Описание», «Иконка».
8. Сущность «**Привязанный облачный сервис**» с атрибутами «Дата окончания», «Токен обновления», «Токен доступа».
9. Сущность «**Библиотека**» с атрибутами «В избранном», «Сыграно», «Последний запуск».
10. Сущность «**Сохранение**», c атрибутами «ИД файла», «Размер», «Дата последней синхронизации», «Название файла», «Описание», «Хеш», «Версия».
11. Сущность «**Пост-Картинка**», связывающая сущности «Картинка» и «Пост».

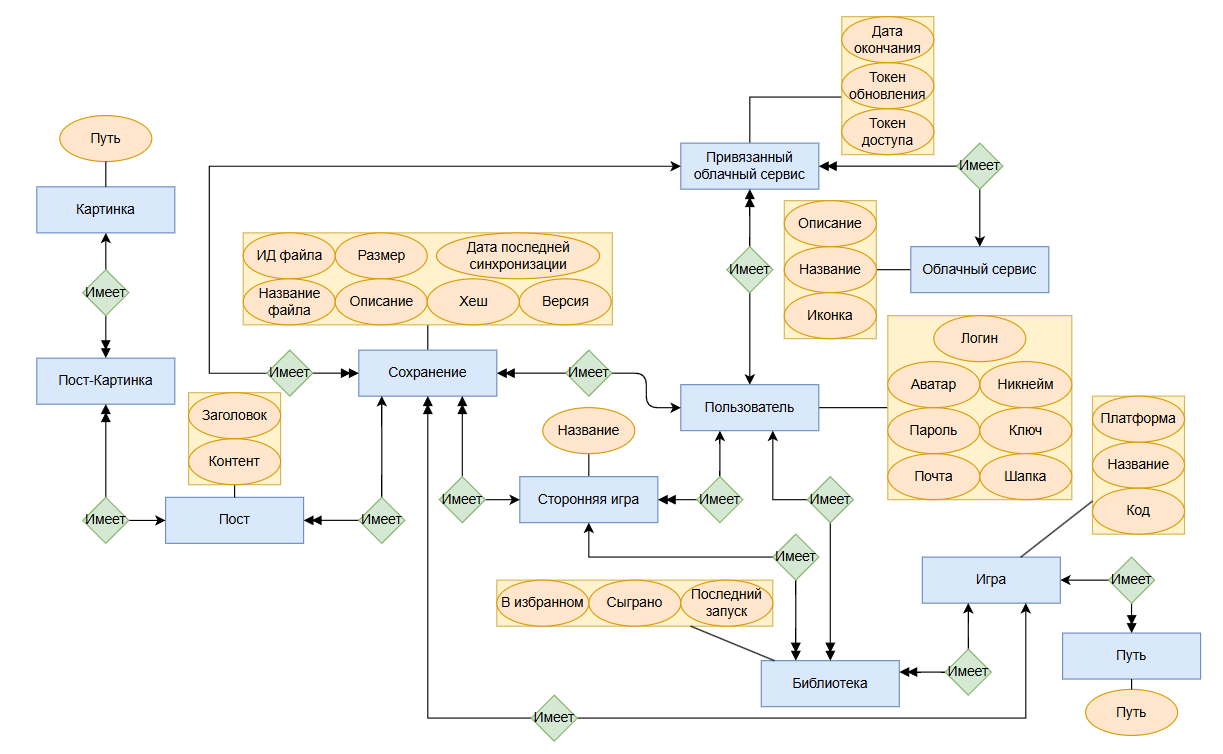


Рисунок 28 – Концептуальная модель данных

## Логическая модель данных

Логическая модель базы данных — схема базы данных, выраженная в понятиях модели данных. Этим отличается от концептуальной модели, описывающей семантику предметной области без указания технологии (конкретных методов реализации), и от физической модели, которая описывает конкретные физические механизмы, применяемые для хранения данных в накопителях.

Используя нашу концептуальную модель данных и выполнив нормализацию данных до 3 нормальной формы включительно, построим логическую модель данных (см. Рисунок 35):

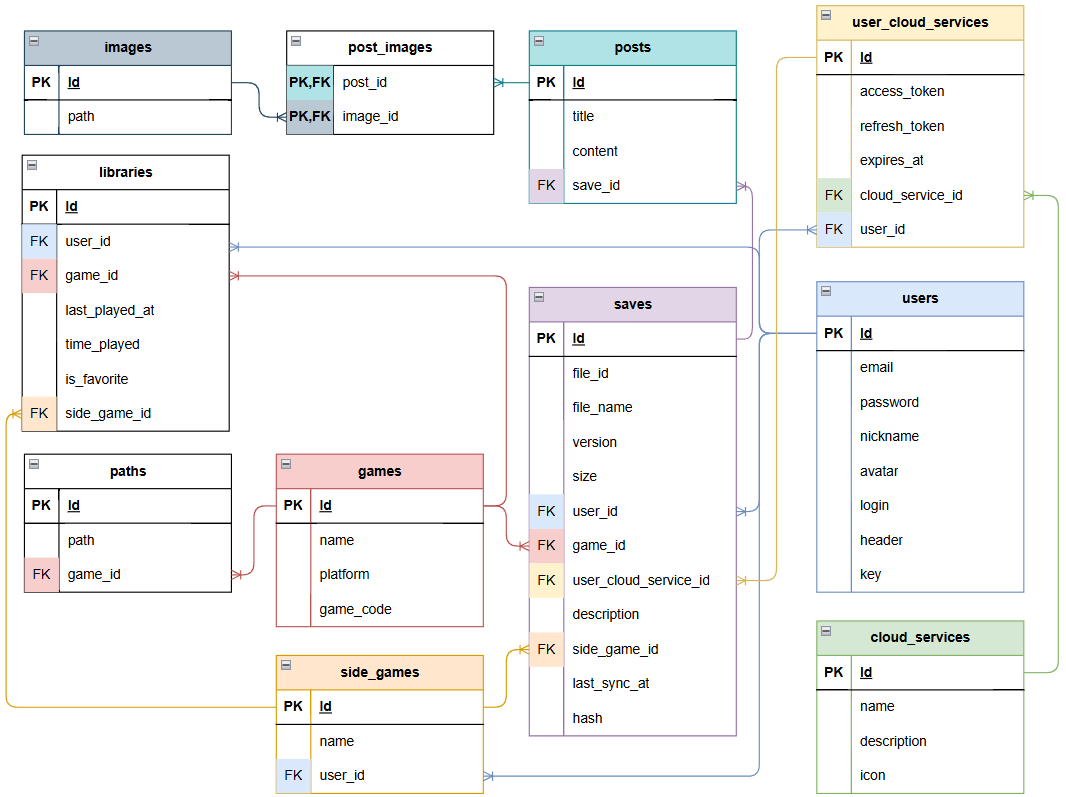


Рисунок 29 – Логическая модель данных

## Физическая модель данных

Физическая модель данных — это модель данных, описанная с помощью средств конкретной системы управления базами данных (СУБД). Физическая модель данных строится на базе логической путем добавления особенностей конкретной СУБД. К таким особенностям могут относиться поддерживаемые СУБД типы данных, соглашения о присвоении имен таблицам, атрибутам и т.д. Физическая модель данных фактически является готовым заданием на создание БД, имея которое можно реализовать БД в выбранной СУБД.

Выбранной СУБД является MySQL. Используя логическую модель, построим физическую модель данных (см. Рисунок 36):

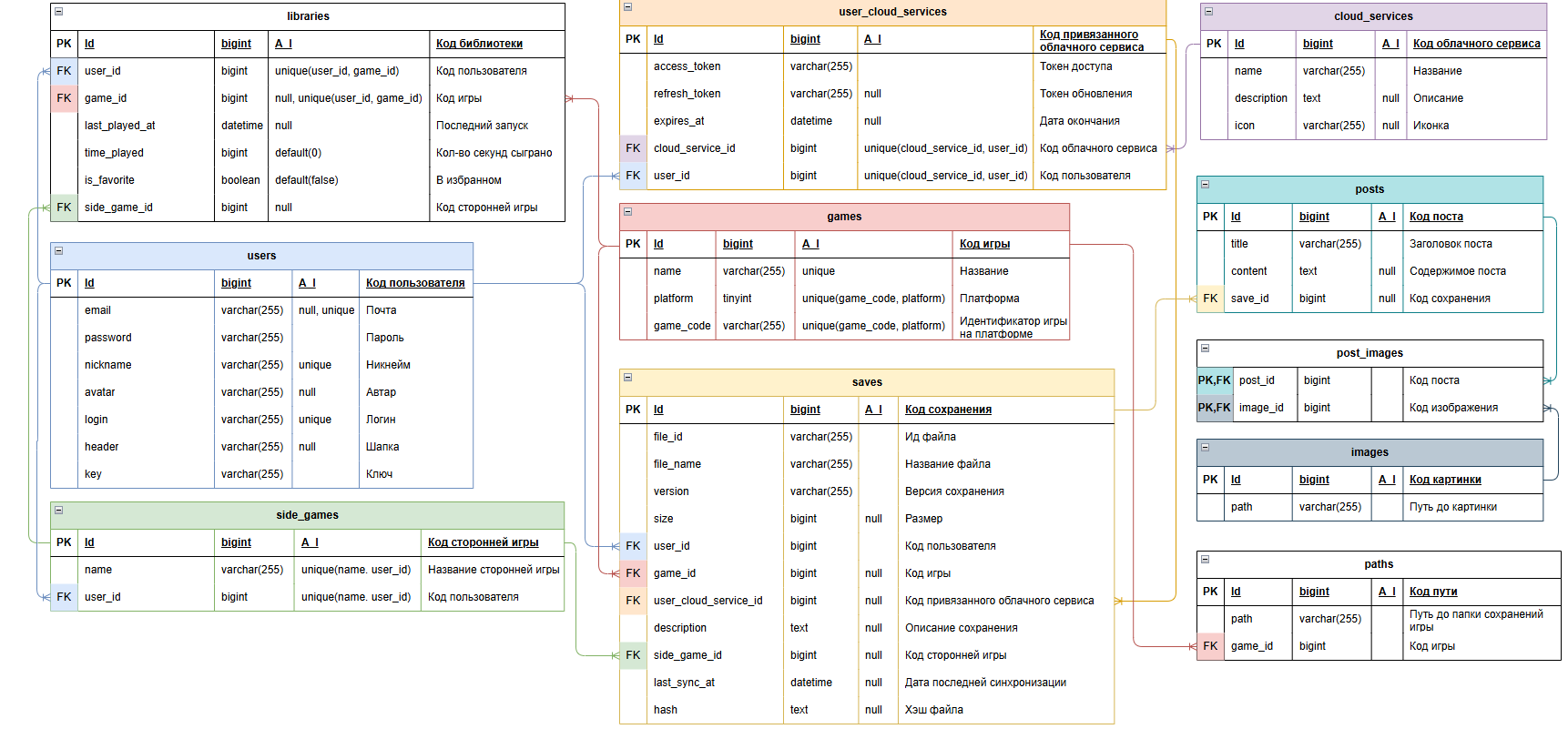


Рисунок 30 – Физическая модель данных

## Словарь данных

Таблица 2 – Словарь данных для БД «PlaySaveBack»

|  |  |  |  |
| --- | --- | --- | --- |
| **users** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | bigint | PK, AUTO\_INCREMENT | Идентификатор пользователя |
| login | varchar(255) | UNIQUE, | Логин пользователя |
| email | varchar(255) | NULL, UNIQUE | Электронная почта пользователя |
| nickname | varchar(255) | NULL, UNIQUE | Никнейм пользователя |
| avatar | varchar(255) | NULL | Аватар пользователя |
| header | varchar(255) | NULL | Шапка пользователя |
| password | varchar(255) |  | Пароль пользователя |
| key | varchar(255) |  | Ключ восстановления пользователя |
| **side\_games** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | bigint | PK, AUTO\_INCREMENT | Идентификатор сторонней игры |
| name | varchar(255) | UNIQUE(user\_id, name) | Название сторонней игры |
| user\_id | bigint | FK, UNIQUE(user\_id, name) | Идентификатор пользователя |
| **games** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | bigint | PK, AUTO\_INCREMENT | Идентификатор игры |
| name | varchar(255) | UNIQUE | Название игры |
| game\_code | varchar(255) | UNIQUE(game\_code, platform) | Код игры на платформе |
| platform | tinyint | UNIQUE(game\_code, platform) | Код платформы расположения игры |
| **cloud\_services** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | bigint | PK, AUTO\_INCREMENT | Идентификатор облачного сервиса |
| name | varchar(255) | UNIQUE | Название облачного сервиса |
| description | text | NULL | Описание облачного сервиса |
| icon | varchar(255) |  | Путь до иконки облачного сервиса |
| **user\_cloud\_services** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | bigint | PK, AUTO\_INCREMENT | Идентификатор привязанного облачного сервиса |
| access\_token | text |  | Токен доступа |
| refresh\_token | text | NULL | Токен обновления |
| expires\_at | timestamp | NULL | Дата окончания токена |
| cloud\_service\_id | bigint | FK | Идентификатор облачного сервиса |
| user\_id | bigint | FK | Идентификатор пользователя |
| **saves** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | bigint | PK, AUTO\_INCREMENT |  |
| file\_id | varchar(255) | NULL | Идентификатор файла в облачном сервисе |
| file\_name | varchar(255) |  | Название файла сохранения |
| version | varchar(255) | UNIQUE(user\_id, game\_id, version), UNIQUE(user\_id, side\_game\_id, version) | Версия сохранения |
| size | bigint |  | Размер файла сохранения |
| description | text | NULL | Описание сохранения |
| hash | text | NULL | Хэш файла сохранения |
| last\_sync\_at | datetime | NULL | Дата последней синхронизации сохранения |
| user\_id | bigint | FK, UNIQUE(user\_id, game\_id, version), UNIQUE(user\_id, side\_game\_id, version) | Идентификатор пользователя |
| game\_id | bigint | FK, NULL, UNIQUE(user\_id, game\_id, version) | Идентификатор игры |
| side\_game\_id | bigint | FK, NULL, UNIQUE(user\_id, side\_game\_id, version) | Идентификатор сторонней игры |
| user\_cloud\_service\_id | bigint | FK, NULL | Идентификатор привязанного облачного сервиса |
| **libraries** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | bigint | PK, AUTO\_INCREMENT | Идентификатор библиотеки |
| user\_id | bigint | FK, UNIQUE(user\_id, game\_id), UNIQUE(user\_id, side\_game\_id) | Идентификатор пользователя |
| game\_id | bigint | FK, NULL, UNIQUE(user\_id, game\_id) | Идентификатор игры |
| side\_game\_id | bigint | FK, NULL, UNIQUE(user\_id, side\_game\_id) | Идентификатор сторонней игры |
| last\_played\_at | datetime | NULL | Дата последнего запуска |
| time\_played | bigint | NULL | Количество секунд, проведенных в игре |
| is\_favorite | bool |  | В избранном |
| **paths** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | bigint | PK, AUTO\_INCREMENT | Идентификатор пути |
| path | varchar(255) |  | Путь до папки сохранений игры |
| game\_id | bigint | FK | Идентификатор игры |
| **posts** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | bigint | PK, AUTO\_INCREMENT | Идентификатор поста |
| title | varchar(255) |  | Заголовок поста |
| content | text |  | Содержимое поста |
| save\_id | bigint | FK | Идентификатор сохранения |
| **images** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| id | bigint | PK, AUTO\_INCREMENT | Идентификатор картинки |
| path | varchar(255) |  | Путь до картинки |
| **post\_images** | | | |
| **Атрибут** | **Тип** | **Ограничение** | **Описание** |
| post\_id | bigint | PK, FK | Идентификатор поста |
| image\_id | bigint | PK, FK | Идентификатор картинки |